同余

2019年3月11日 19:20

-
- 同余

一. 定义：整数ab除以正整数m的余数相等，称a，b模m同余，记为a ≡ b（mod m）
    1. 对m取余得到的余数为a的数的集合为一个模m的同余类，记作a上划线
    2. 模m的同余类一共有m个，这些同余类的集合构成m的完全剩余系
    3. 1-m中与m互质的数代表的同余类有phi（m）个，构成m的简化剩余系
    4. 简化剩余系关于模m乘法封闭
        i. 证：a，b与m互质，a*b mod m也与m互质，也是m的简化剩余系
    5. 应用：计算a+b或a*b对质数p取模时，可以先让a和b分别对p取模，再计算，计算a-b时，同样可先对ab取模，然后需要加p加到保证大于p，再取模

二. 欧拉函数phi(p)：<=p且与p互质的正整数的数量
    1. 性质：积性：对互质的nm，phi(nm) = phi(n)phi(m)
    2. 容斥原理推出的公式：phi[n]= n*∏(1- 1/pi)（pi是n质因子）
    3. 引理：对任意质数p：
        i. phi[p] = p-1
        ii. phi[i*p] = phi[i]*p（i是p的倍数时）
        iii. phi[i*p] = phi[i]*(p-1)（i不是p的倍数时）

三. 费马小定理：对质数p和任意整数a，a^p ≡ a（mod p）
    1. 推论：a^(p-1) ≡ 1（mod p）
    2. 应用：乘法逆元multiplicative inverse：除以a ≡ 乘以a^(p-2)
    3. 打表阶乘的逆元：
        i. 令f(n) = n!，fi(n) = 1 / n! = f(n)的逆元
        ii. 易知f(n) = n*f(n-1)，fi(n-1) = n*fi(n)
        iii. O(n)地递推1~n的f，O(logp)的用快速幂求fi(n)，再O(n)地递推n~1的fi
        iv. n的逆元 = 1/n = f(n-1)*fi(n)，于是1~n的逆元也有了，同理排列组合
    4. 可以视作欧拉定理的特例，证明见欧拉定理↓

四. 欧拉定理：正整数a，p互质，则a^phi(p) ≡ 1（mod p）
    1. 易知当p为质数时phi(p)=p，此时的欧拉定理即为费马小定理
    2. 引理1：与p互质且<=p的数x中，任两个x的差值乘以a也不能与p有整除关系（gcd(a,p)=1，x的差值不超过m，易知gcd(a乘x的差值,p)也=1）
    3. 引理2：与p互质且<=p的数x中，任一个x乘以a，再取模p，都与p互质（可设ax-kp=r，由裴蜀定理，r是a的因数）
    4. 证明：由引理1知ax对p的余数是两两不同的，再由引理2可知，这phi(p)个（ax mod p）恰能各自对应上phi(p)个x。于是phi(p)个ax累乘 ≡ phi(p)个x累乘（mod p），提取公因式x，证毕
    5. 应用：a，p互质时，任意正整数b，a^b ≡ a^(b mod phi(p))（mod p）

五. 扩展欧拉定理：b>=phi(p)时，a^b ≡ a^((b mod phi(p)) + phi(p))（mod p）

1. 证明:
    i. 对p的质因数pi，令p=pi^ri * s，由欧拉定理pi^phi(s) ≡ 1（mod s），由欧拉函数定义phi(p) = phi(s) * phi(pi^ri)，由于幂的指数相乘恒等于幂的乘方，而1的乘方是1，pi^phi(p) ≡ pi^phi(s) ≡ 1（mod s）
    ii. 于是可令pi^phi(p) = ks+1，则pi^(phi(p)+ri) = k*p + pi^ri，则pi^(phi(p)+ri) ≡ pi^ri（mod p），于是得到了p的质因子pi的phi(p)次方也能有类似欧拉定理的 ≡ 1恒等式（？）
    iii. 对b>=ri，有pi^b = pi^(b-ri) * pi^ri ≡ pi^(b-ri) * pi^(phi(p)+ri) = pi^(phi(p)+b)（mod p），即pi^b ≡ pi^(phi(p)+b)（mod p）
    iv. 对任意质因数pi，都可先得到式iii再利用式ii降幂（详见应用iii），于是得到b>=phi(p)时，a^b ≡ a^((b mod phi(p)) + phi(p))（mod p）
2. 应用：当a，p不互质时，对任意**>=phi(p)**的正整数b，也有降幂公式a^b ≡ a^(b mod phi(p) + phi(p))（mod p）
    i. b<phi(p)时，显然不用取余，直接算即可
    ii. phi(p) <= b < 2phi(p)时，显然恒等
    iii. b <= 2phi(p)时，由于ri <= phi(pi^ri) <= phi(p)，b-phi(p) >= ri，因此可以不断-=phi(p)，直至达到ii的范围。即证明里第iv条的降幂

六. 线性同余方程ax ≡ c（mod b）
1. 引：原方程可改写成普通方程ax+by = c
    i. 由扩展欧几里得：由贝祖定理可知，c是d=gcd(a,b)的整数倍时，才有解
2. 令d=gcd(a,b)，x0y0为一组ax+by ≡ gcd(a,b)（mod m）的解，则原方程的特解为x=c/d*x0,y=c/d*y0；通解为x=c/d*x0+b/d*k,y=c/d*y0+a/d*k
```
inline ll exgcd(ll a, ll b, ll &x, ll &y) {
    if(!b) return x=1, y=0, a;
    ll g=exgcd(b, a%b, x, y);
    ll z=x; x=y; y=z-a/b*y;
    return g; }
```
3. 应用：aP互质（gcd==1）时才有模P意义下的a的乘法逆元（顺带一提，有解时，根据欧拉定理，解一定为a^phi(P)-1 % P）
```
inline ll exinv(int a, int P) {  //用exgcd求a模P的逆元
    ll x,y;
    if(exgcd(a,P,x,y)!=1) return -1;
    else return (x%P+P)%P; }
```
4. 再顺带一提，还有这么一种线性打表递推乘法逆元ax ≡ 1(mod p)的方法
    i. 令余数r=p mod d，则p=p/i*i+r ≡ 0(mod p)
    ii. 两边同除ir得p/i/r+1/i ≡ 0(mod p)，于是得到i的逆元为-p/i/r，这其中的-p/i和r=p mod i可O（1）计算，r的逆元肯定在之前打表打过
```
mi[1]=1;
for(int i=2; i<=n; ++i) mi[i]=ll(p-p/i)*mi[p%i]%p;
```

七. 线性同余方程组
1. 孙子定理Chinese Remainder Theorem
    i. mi互质且x系数均为1的方程组有特解：x=Σ(i=1:n) ai Mi ti
    ii. 令mi的乘积为M，Mi=M/mi，ti ≡ Mi^-1（mod mi）
    iii. 证：Mi ti ≡ 1（mod mi）因此ai Mi ti ≡ ai（mod mi）

iv. 证：Mi ≡ 0（mod mk，k!=i），因此ai Mi ti ≡ 0（mod mk，k!=i）

2. 扩展中国余数定理（和CRT没啥关系）：
   i. 用数学归纳法解mi不一定互质但x系数均为1的方程组：
   ii. 第1个方程的解x1=a1
   iii. 设前k个方程的一个特解为xk，设M = lcm(i=1:n)mi，则前k个方程的通解是xk+i*M
   iv. 于是第k+1个方程等价于xk+t*M ≡ a_{k+1}（mod m_{k+1})

```
ll smul(ll a, ll b, ll p) {      //记得传参时先给ab取余一发p
    ll ans=0;
    for(;b;b>>=1) {
        if(b&1) ans=(ans+a)%p;
        a=(a<<1)%p; }
    return ans; }
ll qmul(ll a, ll b, ll p) {
    a%=p, b%=p;
    ll t=(long double)a*b/p;
    ll ans=a*b-t*p;
    return ans<0 ? ans+p : ans; }
ll excrt(int n) {   //解[0,n)
    ll X, Y, M=m[0], ans=a[0];
    for(int i=1; i<n; ++i) {
        ll A=M, B=m[i];
        ll c=(a[i]-ans%B+B)%B;      //新同余方程的右部
        ll g=exgcd(A,B,X,Y);
        if(c%g!=0) return -1;
        X=smul(X,c/g,B/g);
        ans+=X*M;
        M*=B/g;
        ans=(ans%M+M)%M; }
    return (ans%M+M)%M; }
```

八. 高次同余方程

1. a^x ≡ b(mod P)当P为质数时，可以保证模P意义下可对a任意乘除
   i. Baby Step Giant Step分块：令s=sqrt(P)，复杂度约O（sqrtP）
   ii. 令x=si-t，其中t=si%x，则原方程等价于a^(si) ≡ b*a^i（mod P）
   iii. 先将<b*a^i，i>预处理进哈希表，再固定s遍历i即可

```
int bsgs(int a, int b, int p) {  //a^x=b%p的最小非负x，无解时返回-1
    unordered_map<int,int>hsh;
    a%=p, b%=p;
    int s=sqrt(p)+1;
    for(int i=0; i<s; ++i) hsh[ b*qpow(a,i,p)%p ]=i;
    int as=qpow(a,s,p);
    if(as==0) return b==0? 1: -1;
    for(int i=0; i<=s; ++i) {
        int asi=qpow(as,i,p);
        int t=hsh.find(asi)==hsh.end() ? -1 : hsh[asi];
        if(t>=0&&s*i>=t) return s*i-t; }
    return -1; }
```

2. a^x ≡ b(mod P)当aP不一定互质时，不能直接用BSGS
   i. 可以利用(a^x)/d ≡ b/d（mod P/d）约分bP到aP互质为止再BSGS
   ii. 若约分过程中发现b不能被aP的公因子整除，则直接-1
   iii. 约分d的次数会贡献进答案；约分的过程中b还需要额外除以一个式子，可以用exgcd算其逆元，详见洛谷P4195作者eee_hoho的题解
   iv. 需要特判一下b为1的特殊情况

```
int exbsgs(int a, int b, int p) {       //a^x=b%p的最小非负x，无解时返回-1
    a%=p, b%=p;
    if(b==1) return 0;
    int k=1, cnt=0, d;
    while((d=__gcd(a,p))!=1) {
        if(b%d) return -1;
        p/=d, b/=d, k=ll(a)/d*k%p, ++cnt;
        if(b==k) return cnt;
    }
    int ans=bsgs(a,ll(b)*exinv(k,p)%p,p);
    if(ans>=0) ans+=cnt;
    return ans;
}
```

3. x^2 ≡ n(mod P)当P为奇质数时，可用类似虚数平方的方法求解，类似实数域一元二次方程，答案可能为两不同解，两相同解或无实数解，详见solve()

```
int P; //模数，为了避免传参而放在全局变量

inline ll qpow(ll a,int b) {     //a^b%P
    ll ans=1;
    for(; b; b>>=1,a=a*a%P)
        if(b&1) ans=ans*a%P;
    return ans; }

ll t,tt;      //tt为t的平方。注意！复数类中每次乘法都要用到tt！！
struct CP {  //求解二次剩余专用的魔改复数类
    ll x,y;
    CP(ll x=0,ll y=0):x(x),y(y){}
    CP operator*=(const CP&r) {       //乘以r，模数P为全局变量
        return *this = CP((x*r.x%P+y*r.y%P*tt%P)%P,(y*r.x%P+x*r.y%P)%P);
    }
    CP qpow(int n) {    //n次幂，模数P为全局变量
        CP rt = CP(1,0);
        for(; n; n>>=1) {
            if(n&1) rt *= *this;
            *this *= *this;
        }
        return *this = rt;
    }
};

int cipolla(int n) {       //求x%P=n的一个解，P是奇质数，无解时返回-1
    if(n==0) return 0;
    if(qpow(n,(P-1)>>1)==P-1) return -1;    //无解
    srand(time(0));      //初始化随机数种子
    for(;;) {      //随机找到一个满足break条件的t即可
        t=rand()%P;
        tt=(t*t%P-n+P)%P;
        if(qpow(tt,(P-1)>>1)==P-1) break;
    }
    CP rt=CP(t,1).qpow((P+1)>>1);
    return rt.x;
}

inline void solve() {
    int n; scanf("%d%d",&n,&P);       //注意P是全局变量，之后不传参
    int ans=cipolla(n);
```

```
        int ans2=(P-ans)%P; //此处取模纯粹是为了避免ans为0时ans2为P
        if(ans==-1) return puts("Hola!"), void();
        if(ans2<ans) swap(ans,ans2);
        if(ans!=ans2) printf("%d %d\n",ans,ans2);
        else printf("%d\n",ans);
    }
```

-
- 模板&例题

一. 欧拉函数

1. 通式：phi[n]= n*∏(1- 1/pi)（pi是n质因子）

```
ll euler(ll n) {
    ll ans=n, ed=sqrt(n);
    for(int i=2; i<=ed; ++i)
        if(n%i==0) {
            ans-= ans/i;
            while(n%i==0) n/=i; }
    if(n>1) ans-= ans/n;
    return ans; }
```

2. 线性筛实现预处理
   i. 引理：对任意质数p：
   ii. phi[p]= p-1
   iii. phi[i*p]= phi[i]*p（i是p的倍数时）
   iv. phi[i*p]= phi[i]*(p-1)（i不是p的倍数时）
   v. 例：n为偶数时，phi[2n]=2phi[n]；n为奇数时，phi[2n]=phi[n]；

```
int tot, prm[MN], phi[MN];
void init_phi() {
    phi[1]=1;
    for(int i=2; i<MN; ++i) {
        if(!phi[i]) prm[++tot]=i, phi[i]=i-1;
        for(int i=1; i<=tot; ++i) {
            if(i*prm[j]>=MN) break;
            if(i%prm[j]==0) {
                phi[i*prm[j]]= phi[i]*prm[j];
                break;}
            else phi[i*prm[j]]= phi[i]*(prm[j]-1);
        }
    }
}
```

3. 求ab都<n的最简真分数a/b数量（POJ2478）（UVA12995）
   i. 固定b时，a的数量即为phi[b]，因此phi的前n项和即为答案
   ii. 特殊的，一般令phi[1]=1，而1/1不是最简分数，因此从2开始求和
4. 输出a^b mod m，其中b是两千万位数（P5901）
   i. 先求m的phi值，再扩展欧拉定理。快读b，每新读入一位就判断一次是否大于phi[m]，并取余，取余过的话+=一次phi[m]，然后快速幂即可
   ii. 如果不取余也+=phi[m]的话，大多数也能过，但在诸如b为1，a为2，m为4时还是会WA

二. 同余方程

1. 求最小正g满足x+gm ≡ y+gn（mod l）（P1516）
   i. 令(x-y)+(m-n)g = Kl，C=y-x，A=m-n，于是只需求Ag-Kl=C最小g
   ii. 即exgcd形式的Ag ≡ C（mod l）的最小g，该方程通解为x=c/d*x0

+b/d*k，若exgcd解出的x0非负，则最小正x为(c/d*x0)%(b/d)，否则为(c/d*x0)%(b/d)+b/d

iii. 注意x系数a为负时应当两边同时乘负一，才能保证exgcd正常运行

```
ll a=m-n, b=l, c=y-x;
if(a<0) a=-a, c=-c;
ll d=exgcd(a,b,x,y);
if(c%d) cout<<"Impossible";
else cout<<(c/d*x%(b/d)+b/d)%(b/d);
```

三. 乘方塔例题

1. 无限乘方塔，输出2^(2^(2^(2^……))) mod p（P4139）
   i. 设tower(p)计算2的无限乘方塔%p的值，题目写成上述形式后不难看出，调用qpow（a,b,p），固定a=2，用扩展欧拉定理使b=先递归调用tower(phi[p])再+phi[p]值，直至调用tower(1)时，能返回0

```
int tower(int p) {
    if(p==1) return 0;
    int phip=euler(p);
    return qpow(2, tower(phip)+phip, p); }
```

2. 有限乘方塔，输出b层a mod p，0层为1，1层为a（SP10050）
   i. 类似上一题的递归讨论，但麻烦的是递归b层不能保证次数是>phi(p)的
   ii. 最简单的处理方法是返回qpow值的同时，返回有没有取过模
   iii. 坑：0^0是1，0^0^0是0，0^0^0^0是1……
   iv. 优化：针对固定模数，可预处理phi值
   v. （nanti.jisuanke.com/t/41299）有另一种坑，当模数为1时要输出0

```
struct ST {
    int v;
    bool ge;        //大于等于模数与否
    ST(int v=0,bool g=0): v(v), ge(g) {}
};

ST qpow(ll a, ll b, int p) {
    ll ans=1ll;
    bool ge=0;
    while(b){
        if(b&1)
            ans*=a,
            ge|= ans>=p,
            ans%=p;
        b>>=1;
        if(!b) break;//防止被没有乘到的a更新ge
        a*=a;
        ge|= a>=p,    //ans*取余后的a可能更新不了ge，在这也要更新
        a%=p;}
    return ST(ans,ge); }
ST tower(ll a,ll b,int p) {       //计算b层a取余p的值
    if(p==1) return ST(0,1);   //特判取模1的特殊情况
    if(a==1) return ST(1,0);   //特判不取模1但底为1的特殊情况
    if(b==1) return a<p ? ST(a,0) : ST(a%p,1);    //递归终点
    int phip= euler(p);
    ST ans= tower(a,b-1,phip);//递归计算取余phip后的指数
    if(ans.ge) ans.v+= phip;
    return qpow(a, ans.v, p);
}
```

3. 例题：乘方塔序列，输出区间幂塔值，从上一题魔改一下下即可（CF906D）